Project 1:

Submission deadlines: 5:00 pm, Friday 16th April 2021
Value: **10%** of CITS1401.

To be done individually.

You should construct a Python 3 program containing your solution to the following problem and submit your program electronically on Moodle. The name of the file containing your code should be your student ID e.g. **12345678.py**. No other method of submission is allowed. Your program will be automatically run on Moodle for sample test cases provided in the project sheet if you click the "check" link. However, your submission will be tested thoroughly for grading purpose after the due date. Remember you need to submit the program as a single file and copy-paste the same program in the provided text box. You have only one attempt to make the submission so don't submit if you are not satisfied with your attempt. All open submissions at the time of deadline will be automatically submitted. There is no way in the system to open the closed submission and reverse your submission.

You are expected to have read and understood the University's guidelines on academic conduct. In accordance with this policy, you may discuss with other students the general principles required to understand this project, but the work you submit must be the result of your own effort. Plagiarism detection, and other systems for detecting potential malpractice, will therefore be used. Besides, if what you submit is not your own work then you will have learnt little and will therefore, likely, fail the final exam.

You must submit your project before the submission deadline listed above. Following UWA policy, a late penalty of 5% will be deducted for each day (24 hours), after the deadline, that the assignment is submitted. No submissions will be allowed after 7 days following the deadline except approved special consideration cases.

Overview

The year 2020 will be regarded as a pandemic year in the history of mankind. COVID-19 impacted the entire world in such a manner that no other virus has ever done in the history. It's been more than a year for the virus and still uncertainties are looming all over the world.

Center for Systems Science and Engineering at John Hopkins University is regularly gathering the data about the COVID-19 spread and publishing it regularly at https://ourworldindata.org/coronavirus-source-data. This data is sourced from governments, national and subnational agencies across the world and publicly available for researchers and analysts.

In this project, you are required to write a computer program which can read the data from a csv (comma separated values) file provided to you and return different statistical aspects of the COVID-19 cases. Your program should follow the following specifications.

Specification: What your program will need to do

Input:

Your program must define the function **`main`** with the following signature:

```
def main(csvfile,country,type):
```

The input arguments are:

- `csvfile` is the name of the CSV file containing information and record of the COVID-19 cases which needs to be analysed. The first row of the CSV file will contain the headers. From the second row, the first value of each row contains the country code "iso_code", the second and third values will contain the name of continent and country while the fourth, fifth and sixth values contain the date of the year, reported COVID-19 confirmed cases and reported deaths due to COVID-19 respectively. We do not have prior knowledge about the number of countries or days of data available in the CSV file.
- `country` is the country or countries for which we are looking to analyse the record. This input argument will accept a string as the name of country if the third input requires statistical analysis of a particular country. Otherwise this input argument will contain a list of two strings containing two names of countries for which correlation is required.
- `type` is the input argument which mentions which type of analysis are required. It can take only one of the two string inputs: "`statistics`" or "`correlation`". If third input argument is "`statistics`", then the objective of the program is to find the statistical analysis of a single country. Otherwise if the third input argument is "`correlation`" then the objective of the program is to find the correlation of statistical data of two countries.

Output:

The function is required to return the following outputs in the order provided below:

- List containing the **minimum** recorded positive cases of COVID-19 greater than zero for each month of the year for the `country` provided as a second input argument if third input is "statistics". Otherwise if the third input parameter is "correlation", then the output should be a value which is the correlation of minimum recorded positive cases of COVID-19 for each month of the year for the two countries (provided as a single list in second input argument).
- List or value similar to above containing the **maximum** recorded positive COVID-19 cases.
- List or value similar to above containing the **average** recorded positive COVID-19 cases.
- List or value similar to above containing the **standard deviations** in recorded positive COVID-19 cases.

All returned lists should have values recorded for each month of the year in order from January to December. All returned output variables must contain numerical values rounded to four decimal places (if required to be rounded off). Remember not to round the values during calculations and round them only at the time of saving them in the output variables.

Example:

Download the `Covid-data-for-project_1_sample.csv` file from the folder of Project 1 on LMS or Moodle. An example interaction are:

```
>>> mn1,mx1,avg1,std1 = main('Covid-data-for-project_1_sample.csv', "France ",
"statistics ")
```

The output returned in the variables are:

```
>>> mn1

[254, 0, 1032, 633, 140, 136, 4, 3, 4266, 5639, 4354, 3093]

>>> mx1

[41373, 0, 7629, 50746, 4136, 4360, 2488, 11203, 16104, 73010, 106091, 26514]

>>> avg1

[16319.8, 0, 2984.125, 5468.1, 741.6774, 727.5667, 741.7742, 3029.7097, 9491.1,
26079.7097, 31822.0667, 12928.7742]

>>> std1

[11572.5977, 0, 1588.9028, 9377.2189, 850.938, 860.676, 590.4125, 2379.544,
3281.9053, 14927.8697, 24244.2979, 6377.2288]
```

```
>>> mn2,mx2,avg2,std2 = main('Covid-data-for-project_1_sample.csv',
["france","italy"],"correlation")
```

The output returned in the variables are:

```
>>> mn2

0.4013

>>> mx2

0.8827

>>> avg2

0.893

>>> std2

0.8266
```

Assumptions:

Your program can assume a number of things:

- Anything that is meant to be a string (i.e. header row) will be a string, anything that is meant to be date will be a date in the format day/month/year, and anything that is meant to be numeric (i.e. data) will be numeric.

- The string data needs to be considered as case insensitive for inputs parameter as well as inside the file. For instance, the country `"France"` can be provided as input parameter or available in the file as `"France"` or `"france"` or `"FRANCE"` or `"FRanCE"` or any similar way.
- The order of columns in each row will follow the order of the headings provided in the first row. The rows are in random order and their number are not constant.
- No data will be missing in the csv file which means all rows will have complete data.
- It is not mandatory that COVID-19 data is recorded for all days of the month or year. Therefore, while finding average or standard deviation for record of a month, you are required to not to make assumptions about the missing days' data.
- If there is no recorded COVID-19 data for a month then consider it to be zero. The minimum will also be considered as zero which cannot be the case otherwise.
- The **`main()`** function will always be provided with valid input parameters.
- The formula for standard deviation and correlation can be found at the end of the project sheet.

Important grading instruction:

You will have noticed that you have not been asked to write specific functions. That has been left to you. However, it is important that your program must defines the top-level function `main()`. The idea is that within `main()`, the program calls the other functions. (Of course, these may call further functions.) The reason this is important is that when your program is tested, the testing program will call your `main()` function. So, if you fail to define `main()`, my testing program will not be able to test your program and your submission will be graded zero. Don't forget the submission guidelines provided at the start of the project sheet.

Things to avoid:

There are a few things for your program to avoid.

- You are not allowed to import any Python module. While use of the many of these modules, e.g. csv or math is a perfectly sensible thing to do in a production setting, it takes away much of the point of different aspects of the project, which is about getting practice opening text files, processing text file data, and use of basic Python structures, in this case lists and loops.
- Do not assume that the input file names will end in .csv. File name suffixes such as .csv and .txt are not mandatory in systems other than Microsoft Windows.
- Ensure your program does NOT call the `input()` or `print()` functions at any time. That will cause your program to hang, waiting for input that automated testing system will not provide. In fact, what will happen is that the marking program detects the call(s), and will not test your code at all which may result in zero grade.

Submission:

The name of the file containing your code should be your student ID e.g. **12345678.py**. Submit your solution before the deadline electronically on Moodle. No other method of submission is allowed. Your program will be automatically run on Moodle for sample test cases provided in the project sheet if you click the "check" link. However, your submission will be tested thoroughly for grading purpose by teaching team after the due date. Remember you need to submit the program as a single file and copy-paste the same program in the provided text box. You have only one attempt to make the submission so don't submit if you are not satisfied with your attempt. You are encouraged to keep your attempt open as there is no way in the system to open the closed submission and reverse your submission. All open submissions at the time of deadline will be automatically submitted. Separate submission system will be made available for submissions after due date for special consideration or late submissions.

You need to contact unit coordinator if you have special considerations or making submission after the mentioned due date.

Marking Rubric:

Your program will be marked out of 30 (later scaled to be out of 10% of the final mark).

22 out of 30 marks will be awarded based on how well your program completes a number of tests, reflecting normal use of the program, and also how the program handles various states including error states including different number of rows in the input file or missing data for months/days. You need to think creatively what your program may face. Your submission will be graded by data files other than the provided data file. Therefore you need to be creative to look into corner or worst cases. I have provided few guidelines from ACS Accreditation manual at the end of the project sheet which will help you to understand the expectations.

8 out of 30 marks will be awarded on *style* (5/8) "the code is clear to read" and *efficiency* (3/8) "your program is well constructed and runs efficiently". For style, think about use of comments, sensible variable names, your name at the top of the program, etc. (Please watch lectures, where this is discussed.)

Style Rubric:

0	Gibberish, impossible to understand
1-2	Style is really poor or fair
3-4	Style is good or very good, with small lapses
5	Excellent style, really easy to read and follow

Your program will be traversing text files of various sizes (possibly including large csv files) so try to minimise the number of times your program looks at the same data items.

Efficiency Rubric:

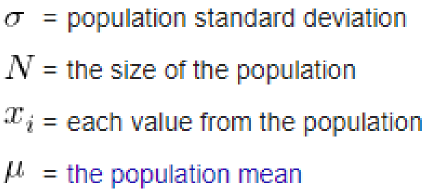
0	Code too incomplete to judge efficiency, or wrong problem tackled
1	Very poor efficiency, additional loops, inappropriate data reading or use of `readline()`
2	Acceptable or good efficiency with some lapses
3	Excellent efficiency, should have no problem on large files, etc.

Automated testing is being used so that all submitted programs are being tested the same way. Sometimes it happens that there is one mistake in the program that means that no tests are passed. If the marker is able to spot the cause and fix it readily, then they are allowed to do that and your - now fixed - program will score whatever it scores from the tests, minus 4 marks, because other students will not have had the benefit of marker intervention. Still, that's way better than getting zero. On the other hand, if the bug is too hard to fix, the marker needs to move on to other submissions. Remember, no extra fixes will be accepted after submission.

Formula:

Standard deviation is mathematically expressed as:

$$\sigma = \sqrt{\frac{\sum(x_i - \mu)^2}{N}}$$



σ = population standard deviation

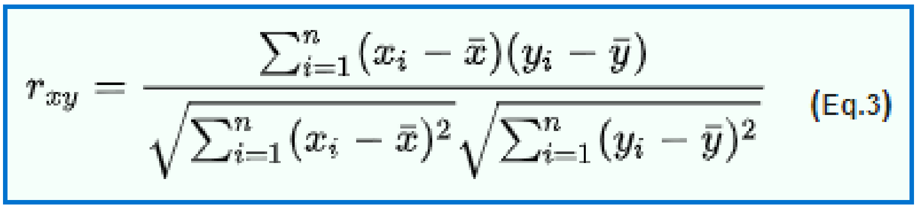
N = the size of the population

x_i = each value from the population

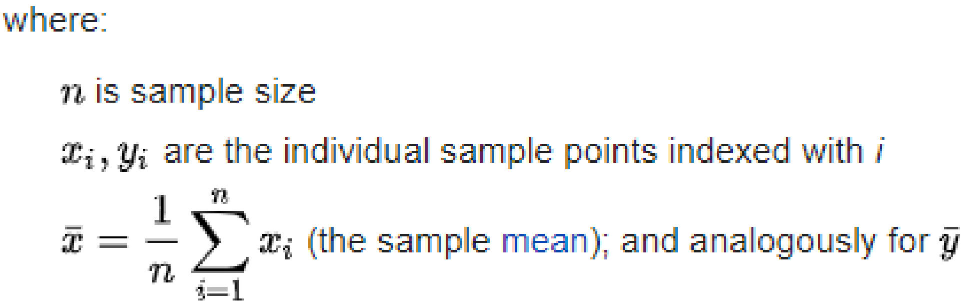
μ = the population mean

You can find more details at https://en.wikipedia.org/wiki/Standard_deviation

The correlation r_{xy} for paired data $\{(x_1,y_1),\ldots(x_n,y_n)\}$ consisting of n pairs, is mathematically expressed as:



$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \quad \text{(Eq.3)}$$



where:

n is sample size

x_i, y_i are the individual sample points indexed with *i*

$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ (the sample mean); and analogously for $\bar{y}$

Extract from Australian Computing Society Accreditation manual 2019:

As per Seoul Accord section D,

A complex computing problem will normally have some or all of the following criteria:

- involves wide-ranging or conflicting technical, computing, and other issues;
- has no obvious solution, and requires conceptual thinking and innovative analysis to formulate suitable abstract models;
- a solution requires the use of in-depth computing or domain knowledge and an analytical approach that is based on well-founded principles;
- involves infrequently-encountered issues;
- is outside problems encompassed by standards and standard practice for professional computing;
- involves diverse groups of stakeholders with widely varying needs;
- has significant consequences in a range of contexts;
- is a high-level problem possibly including many component parts or sub-problems;
- identification of a requirement or the cause of a problem is ill defined or unknown.